Keeping an Eye on Your Security Through Assurance Indicators

Moussa Ouedraogo[1], Chien-Ting Kuo[2], Simon Tjoa[3], David Preston[4], Eric Dubois[1], Paulo Simoes[5] and Tiago Cruz[5]

[1]*Public research Centre Henri Tudor, Luxembourg 1855, Luxembourg*
[2]*Department of Electrical Engineering, National Taiwan University, 106 Taipei, Taiwan*
CyberTrust Technology Institute, Institute for Information Industry, 105 Taipei, Taiwan
[3]*St. Poelten University of Applied Sciences, Matthias Corvinus-Straße 15, A-3100 St. Poelten, Austria*
[4]*School of Architecture, Computing and Engineering, University of East London, London, U.K.*
[5]*CISUC-DEI, University of Coimbra, Coimbra, Portugal*

Keywords: Security Assurance, Verification of Security, Security Management.

Abstract: Despite the incommensurable effort made from across computer sciences disciplines to provide more secure systems, compromising the security of a system has now become a very common and stark reality for organizations of all sizes and from a variety of sectors. The lax in the technology has often been cited as the salient cause of systems insecurity. In this paper we advocate the need for a Security Assurance (SA) system to be embedded within current IT systems. Such a system has the potential to address one facet of cyber insecurity, which is the exploit of lax within the deployed security and its underlining policy. We discuss the challenges associated to such an SA assessment and present the flavor of its evaluation and monitoring through an initial prototype. By providing indicators on the status of a security matter that is more and more devolved to the provider as it is the case in the cloud, the SA tool can be used as a means of fostering better security transparency between a cloud provider and client.

1 INTRODUCTION

The complexity of today's information systems make it difficult for most users to properly identify, assess and address information security risks. Although various security awareness campaigns such as European Cyber Security Month (ENISA, 2014) or National Cyber Security Awareness Month (DHS, 2014) try to increase the level of security awareness, many users still have a wrong risk perception. Users tend to overestimate the security of their systems and often consider security risks as resolved the very moment a mechanism (e.g. anti-malware software, firewall solutions) is put in place. Although these mechanisms inevitably enhance the state of security, the feeling of being protected can lead to risky behaviour such as downloading files from unknown or insecure sources, such as file-sharing websites. If the situation was confined to only the home users, one could put it down to the lack of understanding and expertise in security. However, Loske et al.(2013) concluded in their research that unrealistic optimism about information security risks also takes place at enterprise level. Besides the afore-mentioned challenges of risk perception, a professionalization and states sponsorship of cybercrime (Contreras et al., 2013) create a new cyber threat landscape for organizations. For instance, in 2011 the security of over forty million people worldwide was compromised following a massive network intrusion in the cryptography firm RSA, which resulted in the theft of information related to the SecurID tokens, and the personal information of 102 million registered users of Sonys PlayStation Network (PSN). Recent vulnerabilities, such as Heartbleed or Apples SSL also reveal the vulnerability of security mechanisms we trust. It is true that the reasons we experience security breaches cannot be exhaustively discussed and often this is beyond the understanding of the common user. But why has the security situation exacerbated despite the advances made in the technology for security? In our opinion, three main reasons may explain this. Firstly, our own complacency about security is sometimes to blame. Arbaugh and Frincke (2011), note that mission always trumps security. Therefore, security is often

Ouedraogo M., Kuo C., Tjoa S., Preston D., Dubois E., Simoes P. and Cruz T..
Keeping an Eye on Your Security Through Assurance Indicators.
DOI: 10.5220/0005118504760483
In *Proceedings of the 11th International Conference on Security and Cryptography* (SECRYPT-2014), pages 476-483
ISBN: 978-989-758-045-1

bypassed or turned off as soon as it complicates the completion of an (important) task.
Secondly, although the threat landscape is known to be perpetually changing, deployed security mechanisms are often not maintained to address this issue. In fact even at an enterprise level risk assessments are often conducted as one-off activities. Keeping a security mechanism's deployment features adequate is relevant since the assumption on which its effectiveness level is estimated is strongly linked to the fact that the implementation and/or deployment should conform to certain principles. Any violation of those principles stemming from either a human error or a malicious alteration would result in the security mechanism being ineffective.
Finally, technology does not sit still (Furnell, 2009), nor does the effort to compromise it. In fact, the patching-cracking race between researchers within the field of security and attackers is one where it is increasingly difficult to gain an advantage. This implies that today's state-of-the-art protection is likely to be by-passed with relative ease in the near future, as attackers techniques become more and more sophisticated. The latest Snowden saga has shed light to an even more obscure side of cyber security, whereby it is even harder to distinguish between the "good and bad guys". Given all the above, will we get ahead of this vicious cycle or will we learn to live with it and adapt to the everyday risks (Arbaugh and Frincke, 2011)? We argue that a critical part of living with insecurity resides in the ability to evaluate and monitor the assurance level of the existing security mechanisms and policy through the adoption of a clear strategy. Thus, the current approach, based on proactively identifying threats and mitigating the ensuing risks, needs to be complemented with a more assurance-driven approach to security. In short, security tools such as anti-malware and intrusion detection systems, widely available within current systems, need to be reinforced with Security Assurance (SA) systems with the aim to probe the readiness and correctness of those security mechanisms along with the non-vulnerability of the security policy. In previous publications (Ouedraogo et al. 2012, 2013) we have presented the SA evaluation methodology and metrics. This paper's contribution lies on the analysis of the practical challenges associated to the assessment of SA and the highlight of an SA assessment prototype. It also put in perspective the idea of full SA assessment which will include probing the security policy against known attack patterns.

2 RELATED WORK

Prior related work on Security Assurance has been mainly definitional or focused on the development of metrics relating to SA. ISO/IEC 15408 or Common Criteria (ISO/IEC, 2009) is the most recognised standard in the area. The CC describes a framework in which developers can specify their security requirements, and testing laboratories can evaluate the products to determine if they actually meet the claimed security levels. A number of notable works on security metrics have also been proposed. The Security Metrics Guide for Information Technology Systems NIST 800-55 (Chew et al., 2008) provides guidance on how an organisation, through the use of metrics, identifies the adequacy of in-place security mechanisms, policies, and procedures. The Relative Attack Surface Metrics (Manadhata and Wing, 2011) draw a link between the size of a systems attack surface and the size of the system. Vaughn et al.(2003) proposed a taxonomy organized into two distinct categories of metrics. The first category of metrics (organisational security) aims to assess the Information Assurance (IA) posture (i.e. the actual state) of an organisation while the second category assesses the IA capabilities of a product or system (Technical Target of Assessment TTOA). The Bugyo and Bugyo Beyond projects (Kanstrén et al., 2010) have provided a pioneering initiative to evaluate the SA of systems with emphasis on correctness and the evolving nature of the system model. A critical survey on the evaluation of the SA of operational systems is provided by Hecker (2009).
This paper provides an analysis of the practical challenges to address when engaged in SA assessment. It also presents an initial SA tool along with an analysis of requirements for further improvement.

3 THE CHALLENGES ASSOCIATED TO ITS ASSESSMENT

SA is commonly defined as the ground for confidence that the security mechanisms meet their objectives. A prerequisite and challenge in seeking to evaluate the SA of systems is to identify the key concepts of the security and/or the system as a whole that ought to be assessed in order to develop assurance indicators. Only after that can one start to address how measurements of these concepts can be integrated and communicated to the stakeholders to

ensure a good understanding of the security posture. In order to foster any kind of confidence in the security of a system, there has to be evidence to support the fact that the security-enforcing mechanisms identified as necessary for thwarting a given risk are present (availability) and have been rightly implemented (correctness); and that (ii) they are effective enough to meet the stated security objectives (Effectiveness).

McHugh (Skroch et al., 2000) in the panel workshop on Information Assurance metrics, reflects on security metrics that could provide an indication of the resources or effort required to compromise a system, i.e. a measure of its security effectiveness. McHugh asserts that: For such a measure to be useful, it should be soundly based on demonstrable assurances of the form If I do X (at cost Y) the cost of compromising the system will be raised by Z. Although applicable to security mechanisms such as properly managed crypto-systems, this sort of metric are limited by the fact they do not account for the profile of the adversary, or for the computation power he/she has at its disposition, which can further ease the process of cracking encryptions. Penetration testing, as a means of assessing a network vulnerability can be valuable when it is carried out in the actual operational environment. However as real-world payloads cannot often be used during the development phases, it is difficult to determine the direct effect of penetration testing to effectiveness of the security mechanism. Actually, the testing environment should be as close as possible to the actual operational situation of the system, whether it is a part of an organisation Information System or used by an end-user

Another approach commonly used to identify the strength of the security mechanisms is resorting to security incident records such as log files during a system audit. The assumption being that the number of successful attacks is negatively correlated to the strength of the security mechanisms. Of course, a system that has experienced a plethora of successful attacks may simply not be protected up to the adequate level. However the lack or negligible number of successful attacks may be related to "luck" meaning here the system not having been the target of sophisticated attacks. Since luck remains a concept that cannot be objectively measured, incident records cannot be a reliable source for appraising the strength of one's security. In a similar vein, Martinez-Moyano et al. (2008) have looked into the risk propensity within an organization and how the occurrence rate of risks may elude system managers and thus foster a false sense of trust in their security. The conclusion of their work is that a low level of malicious activity lowers the organization or individual's perceived risk, thereby decreasing its desired investment in security and culminating in even lower levels of detection. In such a situation, the advantage goes to the patient attacker, who can afford to wait until security becomes so lax that a well-planned and properly timed attack can be launched.

4 VALUATION OF SECURITY ASSURANCE

Arguably, users with security expertise or not, care more about effectiveness of the security mechanisms. Therefore, providing them with any indicators on that specific aspect is what they would understand the most. However, measurement of effectiveness as we have discussed in the previous section is not straightforward, nor is it definitive. Moreover, effectiveness is hardly achievable without adequate correctness, which is a focal point in the eyes of the security expert (and increasingly in the eyes of auditors and executives). Reasons for this include the increasingly large demand for techniques for ensuring compliance, which is driven by the prospect of drastic implications that may result in a case of a security breach that can be traced back to a lack of compliance within big corporations (huge fines and compensation penalties, prison sentences, etc.). The lack of correctness alone could be enough for a security expert to grasp the urgency of a situation while meaning little to the less savvy user. For instance, saying to users that their firewalls configuration is such that it allows all incoming connections may be less meaningful than telling them the likelihood of their firewall protecting them against external attack is nearly null. Indeed, a security expert can quickly grasp the urgency of a lack of conformity of a security component for a given system, compared to the common user who hardly knows the policy the security mechanisms are supposedly implementing. In light of this, we advocate a work around strategy to gaining effectiveness: using evidence of security correctness to indirectly appreciate the effectiveness of a security mechanism. The core assumption is that as long as the security mechanisms implementation and deployment conform to up-to-date guidelines or policies and that there are no known vulnerabilities associated with them they can be expected to meet their objective in providing system protection.

In practice, availability of a security mechanism can be verified through the use of an embedded agent running on top of the security mechanism to monitor its operating messages, like sending a beacon packet to an external checking server to verify the security mechanism is running. Through this availability checking method, one could continuously calculate the Mean Time Between Failures (MTBF) and the Mean Time To Recovery (MTTR) as two availability verification indicators of the running security mechanisms and log the operating event of the security mechanisms. A high availability security mechanism will translate into a high MTBF and low MTTR. Properly combined with information on incident records, information about the MTBF of a security mechanism can be used by the security administrator to further justify investment on the latter; especially if an incident can be shown to have occurred while the security mechanism was not operational. Similarly the comprehensive comparison between the MTTR value and the attack event timestamp could help in emphasizing the importance of an information security issue, and support the request for more investment on security mechanisms infrastructure, such as auxiliary redundant systems for enhancing the organization information security. However, if such solutions can be easily implemented on open source security mechanism software by skilfully integrating the agent module for sending a beacon packet, challenges may exist on its implementation on commercial security mechanism products as the latter would have a customized operating system and protocols that may raise compatibility or scalability issues. To solve this problem, an external sensor can be used instead. The external sensor will send the request packets (e.g. ping packet, HTTP request) with an identifiable message (an identifiable IP, HTTP header, etc.) to the security mechanisms periodically, and record the response information for calculating the MTBF and the MTTR as two availability verification indicators of the running security mechanisms and log the operating event of the security mechanisms.

A correctness check relates to measurements that control the compliance of a security mechanism to a predefined policy. Two types of correctness can be distinguished. Deployment correctness, on one hand, relates to the appropriateness of the security mechanisms configuration while activity correctness, on the other hand, concerns the security mechanism compliance to other procedural policies such as frequency of updating it. Examples of activity correctness include: antivirus should be updated every morning, users password to be changed every 90 days etc. Using a mandatory or recommended configuration file (in the case of deployment correctness) or an activity policy specification file/program (for activity correctness) as reference, a dedicated probe in the system may carry out a comparison to inform whether the security mechanism has a correct posture.

In the event a security mechanism is found to have a deviant status, which is a sign of anomaly, the correctness audit frequency could be dynamically adapted. For instance, the probe auditing activity may be increased and made less sparse in time, or other related security checks may be triggered until the security situation has been resolved. We consider two alternatives for the value of security correctness. The first one is to consider only two possible outcomes for the verification of the security mechanism that are compliant or non-compliant. In this case the configuration file of a security mechanism may be hashed and stored in a dedicated database. A comparison between the hash of the current file configuration and the hash of the reference configuration file stored in the database will enable the detection of any erroneous or malicious modification.

The second alternative is based on the idea that one cannot provide more assurance than its quality. In such a case, the values range of security correctness is normalized into discrete values within [0, QL] with QL a discrete value within [1, 5] as advocated by Ouedraogo et al. (2013). Assignment of a security correctness value to a value within 0 and QL depends on the gravity of the mismatch. Thus we assign the value QL to the verification of the correctness of a deployed security mechanism if the security mechanism posture is found to be fully compliant with the security requirements specification, while value zero 0 will be used to signify either that the compliance is at the lowest possible level or that the mismatch detected is critical for the system. Intermediate states will be assigned a discrete value within]0, QL[and classified depending on their gravity for the system. In addition to availability and correctness, the ultimate aim of the vulnerability monitoring is to ensure best practices in terms of deployment and implementation of a security mechanism are continuously applied. Users may elect to use a system with a set of predefined security mechanisms for its protection. However once the system is deployed, previously unknown errors or vulnerabilities may surface for a given security entity, or environmental assumptions may need to be revised. Furthermore, the effectiveness of most security mechanisms is limited

in time. Today's state-of-the-art protection may be by-passed with relative ease tomorrow as attackers techniques are becoming more and more sophisticated. As a result of operation, feedback could be given that would require the operator to correct the system security model, or redefine its security requirements or environmental assumptions with regards to strengthening the security of the system. To handle that eventuality, the vulnerability check, which is associated to each evaluated security entity, uses a known vulnerability database such as the National Vulnerability Database (NVD, http://nvd.nist.gov/cvss.cfm?version=2), to verify whether any vulnerability has been identified for a specific version of an evaluated protection measure or security relevant parameter. Recommendations on how to overcome such an issue are then taken into account by the operator and will help constitute the new reference against which any future conformity evaluation will be undertaken. This is illustrated in Figure 1 through the adaption of the existing policy. By doing so, the system security requirements are permanently updated and will henceforth present enough quality to face up to potential threats to the system.

5 ARCHITECTURE OF THE SA EVALUATION TOOL

We have developed a prototype for appraising and monitoring SA of IT systems. The overall security assurance system is composed of two main components: the measurement framework and the SA monitoring cockpit, architecturally structured as shown in Figure 2. The assurance monitoring cockpit enables the operator to visualize assurance fluctuation, as well as providing him with relevant information explaining the reasons for the eventual drop in the assurance level. This later aspect can be seen as the assistance part of the system as it supports the assurance evaluator in diagnosing security concerns and eventually in decision making. The cockpit also has a functionality that helps document the assurance levels of components of concern over time. The vulnerability report of the security assurance cockpit, fed by vulnerability databases such as the NVD, enable the security practitioner to visualize information related to possible emerging vulnerability for a component, along with guidelines on how to address them. The measurement framework is concerned with coordinating and gathering information from probes deployed in the system to verify the compliance of a component security measure with respect to a security policy. Given the highly distributed nature of most current systems, we resort to agent technology for the SA assessment. The implementation was conducted on SPADE platform using open source and free software: Python v.2.7 (http://www.python.org/) as programming language, Cherrypy (http://www.cherrypy.org/) for the web service and SQLite (http://www.sqlite.org) for the database. The agents organisation involves a hierarchy of four types of autonomous and collaborative agents, as illustrated in Figure 2: the Security Assurance evaluation manager or Server agents, Security mechanism verification manager or multiplexer agents (referred to as MUX-agents), and the security mechanism verification performer or probe agents. The first level of agents is on the server side; essentially it is a single agent that is embedded within the server. When the server receives a request to perform a Security Assurance evaluation, the server agent handles the request and identifies the appropriate MUX agents (using a role directory).

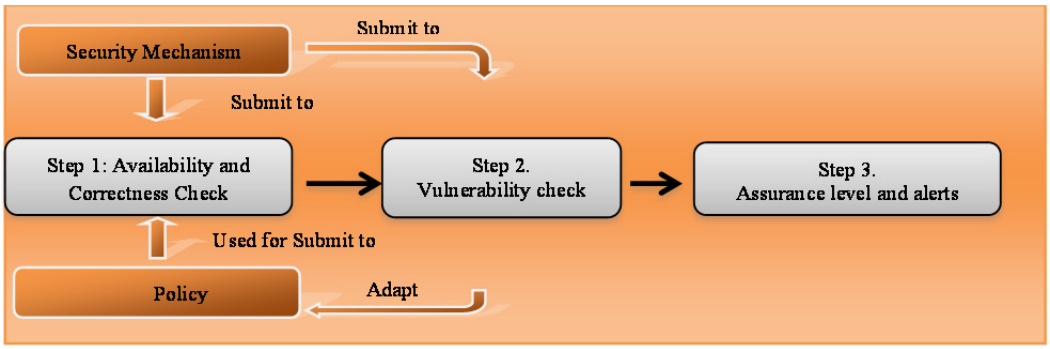

Figure 1: Security assurance strategy.

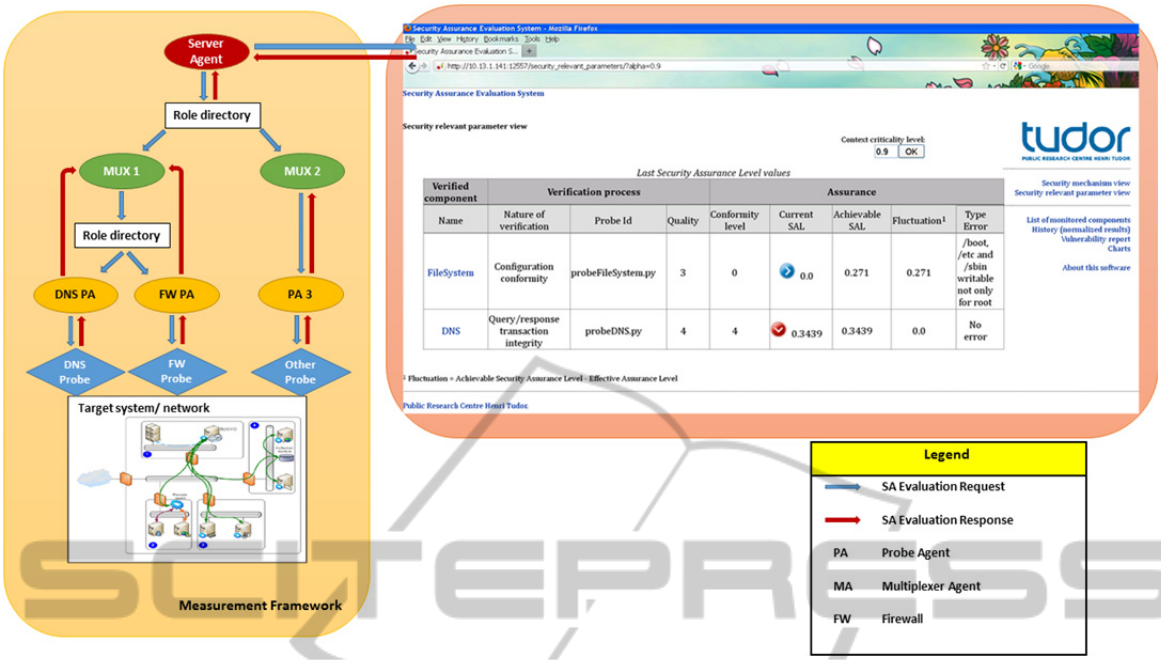

Figure 2: Security assurance evaluation system and architecture.

Upon reception of the request, the MUX agent is responsible for dispatching the verification to perform to the dedicated probe agents that they control. Alternatively, when a verification has been performed and the results are being sent back to the server through MUX-agents, the server agent is the one that processes the information, and aggregates them before they are displayed in a more meaningful form for the system operator. The second type or MUX agents play the role of intermediary between the server agent and probe agents. They supervise the activities of a group of probe agents. MUX agents are meant to forward measurement request from the server agent to the right probe agents using a role directory.

They also collect the results of the measurements coming from the probes. Finally, probe agents trigger their associated probe in the event of receiving a measurement request from a MUX agent. They also collect the measurements result from probes and transmit them to MUX agents. The message format between agents is not unique. In fact an XML based format is used between Server agent and MUX-agents. However the message format between a probe agent and a probe is specific to the probe. Thus the message from the probe agent has to be transformed in a format understandable by the probe.

6 RELATIONSHIP WITH EXISTING EFFORTS AND FUTURE FOR THE SOFTWARE PROTOTYPE

Computational needs for our model depend on (i) the time of use (the database will grow gradually as the probes will return the information) and (ii) the number of probes that will return the information (bigger network, more probes, and more information in the database). Nonetheless, given the type of data stored (essentially floats and strings) the growth in the database size is slow. Moreover SQLite can enable storage of 14124 articles (long string) using 32 Mb. In all, the model uses very low memory and CPU usage, and can run normally on a computer of 4GB of RAM and a 2GHz processor. Nevertheless, our experience with the prototypes operational validation has hinted some issues and areas for future research. In fact it is desirable that the evaluation system is not too intrusive and overloads the system, it is meant to probe and induce a degradation of its Quality of Service (QoS). Therefore any usage of the Security Assurance evaluation system should be conducted in a context that minimises its interference with the evaluated system. Another important consideration is that the transport link between the evaluation framework and

the user/operators monitor should be secured and resilient. Similarly, access to the Security Assurance evaluation system should be controlled. A lack of stringent security could in fact result in a malicious individual gaining information related to the systems security and enable him/her
to wait for an auspicious time to launch an attack. Alternatively the communication links between a user/operators monitor and the evaluation system on the one hand, or between entities within the evaluation framework itself on the other hand, may accidentally break leading to security information black outs.

On top of probing the deployed security, work relating the systematic verification of security policies as another facet of the broader SA assessment initiative is highly important and requires the attention of researchers. In fact the current indicators provided by the SA tool remains as reliable as the security policy since the verifications are conducted taking the later as reference. Thus, because a policy can also be the source of security compromise and a vector for further propagation of a risk within a network, considering it as appropriate without further checks can only provide partial assurance to the system's stakeholders. One way of accounting for the adequacy of policy, which we are currently exploring includes the use of satisfiability tests with conducted by a logic reasoner that will verify whether known attack patterns can be deduced from the policy specification. Finally by providing indicator on the status of a security matter that is often devolved to the provider as it is the case in the cloud, the SA tool can be used as a means of fostering better security transparency between a cloud provider and client.

Our SA assurance tool has a Security Content and Automation Protocol or SCAP capability though currently our checklist related to the security mechanisms is based on instructions and check list derived from reference documents such as security policies that are subsequently put in an XML-based format. The adoption of the XCCDF format is therefore planned to allow it being more compatible with other SCAP tools and thus leading to a standard information interchange, document generation, automated compliance testing, and scoring.

7 CONCLUSION AND FUTURE WORK

In this article we have discussed the challenges relating the evaluation of SA and presented the features of our SA evaluation tool. We have also advocated the use of such a system to better manage security of IT systems. Certainly, addressing SA at the level of operational systems is only one facet of security management. Other important aspects include: the selection of the security mechanisms; their strength and that of the security policy. Nonetheless, without adoption of a rigorous and continuous Security Assurance activity it is hardly possible to guarantee and maintain security regardless of the security level aimed at.

ACKNOWLEDGEMENTS

This work has been conducted in the context of the SAINTS project, financed by the national fund of research of the Grand Duchy of Luxembourg (FNR) under grant number C12/IS/3988336.

REFERENCES

Arbaugh W.A. and Frincke, D.A, 2011. Living with insecurity." *IEEE Security & Privacy, vol. 9, no. 6*, pp. 12–13.

Chew, E, Swanson, M ,Stine, K., Bartol, N., Brown A., and Robinson, W, 2008. Security metrics guide for information technology systems rev.1, *Nist special publication* 800-55: National Institute of Standards and Technology, Tech. Rep., 2008.

Contreras J. L, DeNardis, L. and Teplinsky, M, 2013. Mapping today's cybersecurity landscape," American University Law *Review, vol. 62, no. 5*, p. 1117.

DHS, 2014. National cyber security awareness month. [Online]. Available:https://www.dhs.gov/national-cyber-security- awareness-month

ENISA, 2014. European cyber security month. [Online]. Available: https://cybersecuritymonth.eu/

ISO/IEC (2009), ISO/IEC 15408-1:2009, International Organization for Standardization and the International Electrotechnical Commission, Geneva.

Furnell S.M. (2009) 'The irreversible march of technology', Information Security Technical Report 14(4)pp.176-180, Elsevier.

Hecker A. and Riguidel, M. 2009. On the operational security assurance evaluation of networked it systems," in Smart Spaces and Next Generation Wired/Wireless Networking. *Springer*, pp. 266–278.

Kanstren, T., Savola R. , Evesti, A., Pentikäinen, H., Hecker, A., Ouedraogo, M. , Hatonen, K., Halonen P., Blad, C., Lopez O. 2010. Towards an abstraction layer for security assurance measurements, *in Proceedings of the Fourth European Conference on Software Architecture: Companion Volume. ACM, 2010*, pp. 189–196.

Loske A, Widjaja T, and Buxmann, P. 2013. Cloud computing providers' unrealistic optimism regarding it security risks: A threat to users?" in Thirty Fourth International Conference on Information Systems (ICIS), [Online]. Available: http://aisel.aisnet.org/cgi/viewcontent.cgi?article=1200&context=icis2013

Manadhata P.K. and Wing, J.M., 2008. An attack surface metric, *IEEE Transactions on Software Engineering, vol. 37, no. 3*, pp. 371–386, 2011.

Martinez-Moyano, I, J, Samsa, M,E, Burke, J, F, Akcam B. K. 2008. Toward a generic model of security in an organizational context: Exploring insider threats to information infrastructure, in Hawaii International Conference on System Sciences (HICSS).

Ouedraogo, M, Khadraoui,D., Mouratidis H., and Dubois, E. 2012, "Appraisal and reporting of security assurance at operational systems level", *Journal of Systems and Software, vol. 85, no. 1*, pp. 193–208.

Ouedraogo, M, Savola, R, M, Mouratidis, H. Preston, D. Khadraoui, D and Dubois, E, 2013. Taxonomy of quality metrics for assessing assurance of security correctness, *Software Quality Journal, vol. 21, no. 1*, pp.67–97.

Payne, S, 2006. A guide to security metrics," SANS Institute, InfoSec Reading Room.

Savola R.M, Pentikainen, H and Ouedraogo, M. 2010. Towards security effectiveness measurement utilizing risk-based security assurance", *in Information Security for South Africa (ISSA), IEEE.*

Skroch, M. McHugh, J. and Williams, J. 2000. Information assurance metrics: Prophecy, process r pipedream," in Panel Workshop, National Information Systems Security Conference (NISSC 2000), Baltimore.

Vaughn R. B., Henning, R. and Siraj, A. 2003. Information assurance measures and metrics-state of practice and proposed taxonomy," *in System Sciences. Proceedings of the 36th Annual Hawaii International Conference on. IEEE*.

Waltermire, D. Quinn, S. Scarfone, K. Halbardier, A. 2009. The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.2., *Special Publication 800-126, NIST*.